Generazione automatica di mappe della tessitura dei suoli in GRASS attraverso l'uso di uno script in python (pyTexture_0.5.2).

Gianluca Massei (g_massa@libero.it)

Introduzione

Negli studi a carattere territoriale la conoscenza della tessitura dei suoli è un parametro importante perché riassume alcune proprietà pedologiche fondamentali dal punto di vista pratico e applicativo.
Numerose esperienze italiane ed estere hanno dimostrato la possibilità di costruire mappe dettagliate, a partire da misurazioni puntuali, grazie a tecniche avanzate di geostatistica; per tale ragione, sempre più spesso, nella descrizione di un territorio vengono predisposte anche cartografie della distribuzione spaziale delle principali proprietà pedologiche a cui, però, non segue con altrettanta frequenza la predisposizione di carte della tessitura secondo i principali schemi di classificazione.
Una delle principali difficoltà al riguardo è quella di non disporre di procedure automatiche o semiautomatiche in grado di mantenere il livello di dettaglio raggiunto con le singole mappe della distribuzione di sabbia, limo e argilla. In altri termini, se per classificare un profilo di suolo dal punto di vista della tessitura esiste una procedura deterministica e precisa (attraverso il triangolo della tessitura), trattando un dato spaziale non si ha a disposizione, almeno nei principali e più diffusi software GIS, alcuno strumento in grado di attuare la stessa procedura e fornire un dato in output in modo automatico e attraverso un algoritmo di tipo deterministico.

Obiettivi

Partendo dalle considerazioni espresse in premessa, è stato progettato un percorso finalizzato ad arricchire GRASS di un modulo in grado di elaborare i grid di sabbia e argilla di un determinato territorio e fornire in output il relativo grid della tessitura dei suoli. Per fare ciò, si è scelto di lavorare per step successivi: una prima fase finalizzata alla redazione di uno script in python con il duplice obiettivo di testare l'algoritmo e mettere a disposizione da subito uno strumento che operasse con sufficiente precisione ed efficienza; una seconda fase del lavoro, invece, finalizzata alla "traduzione" dello script in python in codice in C per realizzare un modulo specifico di GRASS, sfruttando tutte le potenzialità di questo software GIS.
Il presente lavoro si concentra principalmente sulla prima fase di descrizione e testing del nuovo script in Python (pyTexture), lasciando alla sola parte conclusiva una breve illustrazione del modulo di GRASS provvisoriamente chiamato "r.soil.texture".
Lo script pyTexture accetta in input i dati derivati da GRSS in formato ASCII GRID, riferiti alla distribuzione di sabbia e argilla, e fornisce in output la relativa carta della tessitura secondo gli schemi di classificazione USDA, FAO e International. Il dato in output è sempre in formato ASCII GRID e può essere importato in GRASS per successive elaborazioni.
Il linguaggio di script utilizzato, Python2.4, consente di gestire efficacemente i file ASCII e di ottenere i risultati voluti attraverso la scrittura di una ridotta quantità di codice, peraltro utilizzabile su qualsiasi piattaforma.
Utilizzando questo primo script è stata eseguita una prima analisi sulla correttezza dell'algoritmo al fine di evidenziare la rispondenza degli output del modulo in esame con quelli provenienti da modelli che lavorano su dati puntuali disponibili e frequentemente utilizzati per la classificazione della tessitura dei suoli.

Materiali e metodi

Python, nella versione 2.4, è ormai molto più di un semplice linguaggio di script, infatti è in grado di supportare applicazioni di rilevante complessità. Rimanendo nel campo della geomatica, basta ricordare che OpenEV e Thrban sono interamente scritti in python e che sono disponibili per lo stesso linguaggo le librerie gdal/ogr nonché le PCL per lo sviluppo di nuove applicazioni. Si caratterizza per la curva di apprendimento piuttosto ripida, associata ad una notevole modernità e continuo sviluppo dello stesso linguaggio.

Python è un linguaggio orientato agli oggetti, scarsamente tipizzato e multipiattaforma; è molto ricco di librerie che possono essere importate ed utilizzate con estrema semplicità nelle applicazioni.

Tutte queste ragioni sono alla base della scelta di utilizzare questo strumento per la prima implementazione dell'algoritmo di definizione della tessitura suoli.

La scelta di utilizzare i GRID di GRASS come file di input e output di pTexture è basata sostanzialmente sulla semplicità della strutture del file e la immediatezza con cui lo stesso può essere trattato con le librerie standard di python. L'uso delle librerie GDAL, pure se implementate in python, avrebbero reso meno indipendente lo script e, soprattutto, meno trasparente il funzionamento dell'algoritmo. Tutto ciò ha come prezzo da pagare la ridotta velocità di elaborazione dei file di dimensioni significative. E' da tenere presente, comunque, che questo problema è solo transitorio perché l'obiettivo finale del progetto è quello di ottenere un modulo di GRASS scritto in linguaggio C che lavori direttamente sui raster binari di GRASS.

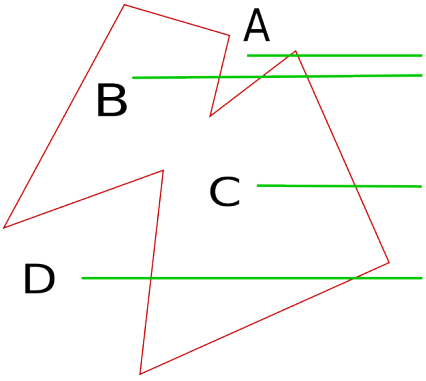
La classificazione della tessitura dei suoli avviene normalmente con l'ausilio di specifici diagrammi triangolari che riportano lungo gli assi i valori percentuali di sabbia, limo e argilla e il cui interno è diviso in classi di tessitura. I valori di granulometrici vanno riportati sugli assi e da ciascuno di questi viene tracciata una retta: dall'incrocio della tre si ottiene uno e un solo punto che ricade nell'area del triangolo che identifica la classe tessiturale ascrivibile al profilo esaminato. In termini bidimensionali, si tratta di ripetere per ogni cella dei grid di sabbia e argilla questa operazione, per ottenere così un nuovo grid con i pixels classificati in funzione della classe tessiturale.

Gli schemi maggiormente utilizzati nella classificazione sono quello dell'USDA, della FAO e International e, pertanto, queste tre possibilità sono implementate nello script e disponibili per l'uso.

Il problema da risolvere, pertanto, si basa sulla necessità di definire se un punto con coordinate note (X_{SABBIA} e $Y_{ARGILLA}$) si pone all'interno o all'esterno di un determinato poligono del triangolo della tessitura in modo che, quando la condizione di appartenenza risulta vera, lo stesso punto viene qualificato con una determinata classe tessiturale.

L'approccio utilizzato per fare ciò è un algoritmo, proposto da R. Franklin, che traccia una semiretta a partire dal punto di coordinate note e misura il numero di intersezioni con i lati dei poligoni posti a destra del punto; quando le intersezioni sono in numero pari il punto è esterno, con intersezioni in numero dispari, invee, il punto è interno.

Figura 1: i punti A e D sono esterni al poligono (intersezioni pari), mentre i punti B e C sono interni (intersezioni dispari) – sul lato destro è riportata l'implementazione in python dell'algoritmo di Franklin utilizzato nello script.



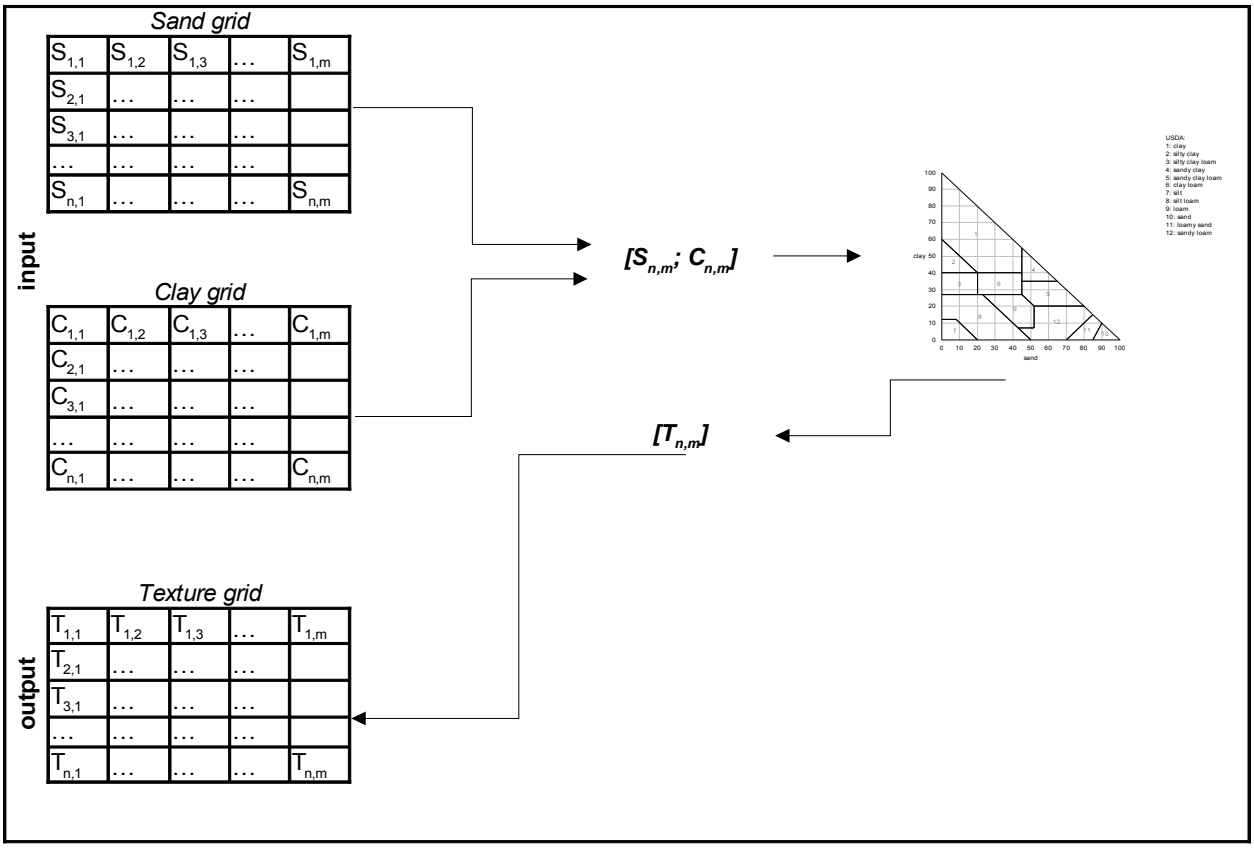

```
def pnpoly(npol, xp, yp, x,y):
        c=0
        j=npol-1
        for i in range(npol):
                if((yp[i] <= y < yp[j] or yp[j] <= y <yp[i]) and
                (x<(xp[j] – xp[i]) * (y-yp[i])/(yp[j]-yp[i])+xp[i])):
                c=not c
                j=i
        return c
```

In sostanza, pyTexture accetta in input due grid (sabbia e argilla) relativi alla stessa porzione di territorio, riferite allo stesso datum cartografico ed aventi uguali dimensione delle celle, estrae da ogni singola cella il valore, ne determina le "coordinate tessiturali" e definisce la classe di tessitura secondo l'algoritmo di Franklin. Sulla base di tale risultato genera un nuovo grid, contenente la perimetrazione delle classi di tessitura rilevate (Fig. 2)
La struttura di *pyTexture* si compone di un set di file che possono essere brevemente descritti come segue:

- un modulo con il codice scritto in python che rappresenta il nocciolo della procedura;
- tre file *.dat contenenti le informazioni che consentono a *pyTexture* di "ricostruirsi" il triangolo della tessitura secondo gli schemi USDA, FAO, International;
- tre file *.sh per riclassificare ed attribuire la label ai files di output dopo l'importazione in GRASS; è, in sostanza, un comando r.reclass che attribuisce la corretta label a seconda dello schema di classificazione impiegato;
- un file readme.txt dove sono riportati informazioni generali e la "storia" sullo script;

L'interfaccia è a linea di comando ed è disponibile un parser unix-like del tipo:
"*python pyTexture_0.5.2.py -s sand -c clay -t USDA -o Tessitura*" il cui utilizzo non dovrebbe essere particolarmente problematico per gli utenti GRASS.

Figura 2: schema semplificato di *pyTexture*



Applicazione di pyTexture ai suoli della Valle Umbra

Per valutare la bontà dell'algoritmo di classificazione implementato, sono stati utilizzati alcuni dati pedologici rilevati direttamente nella Valle Umbra (fig. 3).

Figura 3.a: Inquadramento territoriale e delimitazione dell'area di studio

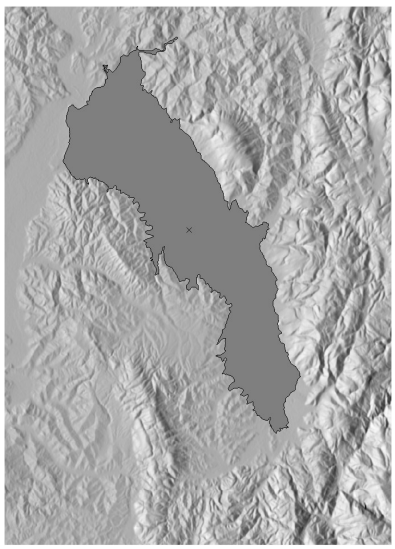


Figura 3.b: Distribuzione dell'argilla nei suoli

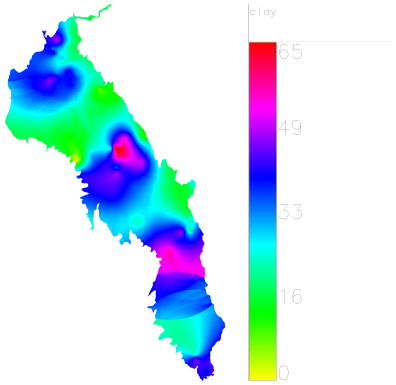
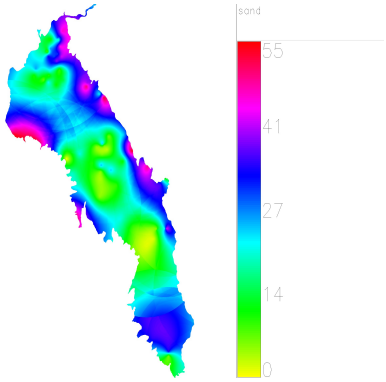
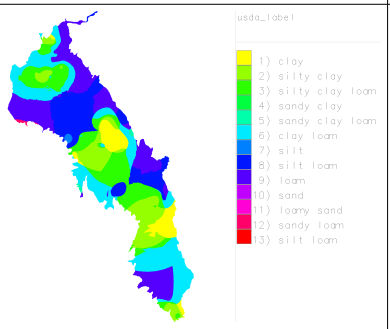
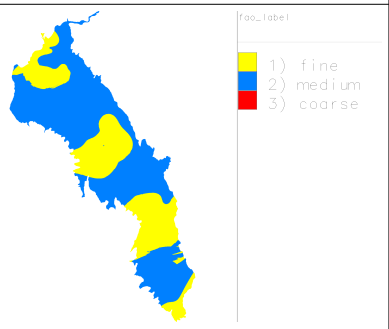
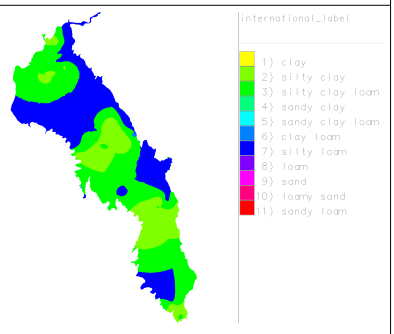


Figura 3.c: Distribuzione della sabbia nei suoli



La porzione di territorio esaminata è caratterizzata da un livello di utilizzazione antropico molto elevato dove l'urbanizzazione, a fini abitativi e produttivi, l'agricoltura intensiva, la viabilità, hanno determinato un livello di vulnerabilità dell'acquifero preoccupante.
I dati pedologici utilizzati sono costituiti da campioni di suoli sui quali è stata determinata la quantità di argilla e di sabbia. Utilizzando tecniche di kriging sferico sono state generate rispettivamente la mappa della distribuzione della sabbia e dell'argilla (Figure 3.b e 3.c).
L'applicazione dello *script pyTexture*, ai file di sabbia e argilla della Valle Umbra, ha generato la corrispondente mappa della tessitura riportate in fig. 4.
Lo schema di classificazione fornisce degli identificativi numerici che contraddistinguono la tipologia tessiturale secondo lo schema del software TAL ver 4.2 (anche questo open source) ma, al fine di evitare confusione, sono stati predisposti tre semplici script che ricostruiscono le label corrette in funzione dello schema di classificazione utilizzato(FAO, USDA, International).

Figura 4.a: mappa della tessitura secondo lo schema USDA.	Figura 4.b: mappa della tessitura secondo lo schema FAO	Figura 4.c: mappa della tessitura secondo lo scema International
		

Verifica dei dati

La procedura di classificazione della tessitura lavora presupponendo che i dati di input (sabbia e argilla) siano corretti, errori nella definizione dei rispettivi grid si ripercuotono inevitabilmente sulla classifica della tessitura e, di contro, una elevata precisione nella procedura di generazione delle mappe si tradurrà in una definizione della tessitura dei suoli particolarmente accurata.
La verifica è stata condotta su 76 punti coincidenti con i profili utilizzati per generare i grid di sabbia e argilla alla base del calcolo della tessitura, ad ognuno è stato attribuito il valore di sabbia e argilla corrispondente alla cella di grid coincidente con il punto, oltre a mantenere i rispettivi valori originali del profilo. In altri termini è stata creata una la tabella di verifica costituita dai seguenti campi: identificativo del profilo, percentuale di sabbia e argilla rilevate direttamente sul suolo, percentuale di sabbia e argilla trasferita al punto dal gird generato con il kriging
Utilizzando il software TAL ver. 4.2 dai dati tabellari descritti sono stati elaborati due ulteriori campi di tessitura, uno relativo a sabbia e argilla misurati sul campo ed uno relativo a sabbia e argilla calcolati per "interpolazione spaziale".
Alla tabella così ottenuta, è stato aggiunto un campo contenente la tessitura del profilo proveniente dalla cella del grid derivato da *pyTexture*. In questo modo è stato possibile identificare le eventuali differenze di classifica della tessitura per uno stesso punto sul terreno utilizzando uno strumento validato e diffuso (TAL ver 4.2) e lo script in esame che lavora sui grid di GRASS. Come riportato nella tabella, vi è una corrispondenza del 97,4% tra i valori di tessitura calcolati con i due procedimenti

Tab.1 – Risultati della verifica dei dati ottenuti con pyTexture e TAL ver 4.2.

	n° campioni	***%***
Non corrispondenze	1	1,3
Corrispondenze parziali	1	1,3
Corrispondenze totali	74	97,4

La corrispondenza parziale deriva dalla possibilità che TAL ha di attribuire una doppia label ad un punto che si pone sulla linea di confine di due diverse classi di tessitura. E' evidente che pyTexture non può fare ciò perché lavora su dati bidimensionali e, pertanto, la corrispondenza si ha solo per una delle due stabilite da TAL.
Con l'elaborazione delle mappe con tecniche di Kriging si hanno dei problemi dovuti alla non corrispondenza esatta tra la dimensione della cella del grid (su cui lavora pyTexture) rispetto a quella puntuale del profilo(oggetto di elaborazione di TAL)
Lavorando ulteriormente nella generazione del dato di input con più adeguate tecniche geostatistiche, è lecito supporre che la coincidenza tra i due modelli possa avvicinarsi ulteriormente al 100%.

Conclusioni

I dati ottenuti dimostrano che vi è una corrispondenza molto elevata tra i dati forniti dai due modelli il che dimostra la correttezza dell'algoritmo e della procedura che lo implementa. Le differenze sono legate alle fasi di passaggio che, in ogni caso, sono difficili da delimitare con un confine netto e, per tale ragione, un prossimo obiettivo potrebbe essere quello di arricchire l'algoritmo con un approccio fuzzy.
Il pyTexture ha il vantaggio di lavorare direttamente con mappe della distribuzione della sabbia e dell'argilla, ormai facilmente calcolabili grazie alle tecniche geostatitiche ed ai relativi software disponibili (come lo stesso GRASS). Naturalmente, il passaggio critico è proprio la determinazione della distribuzione spaziale di tali variabili, poiché, come evidenziato, l'attendibilità della carta della tessitura risulta molto sensibile alla qualità dei dati di input.
Il modulo *pyTexture* può essere richiesto all'autore (g_massa@libero.it) ed è rilasciato secondo i termini della licenza GPL. E' in corso di testing un modulo interno a GRASS 6.0 scritto in C che

implementa lo stesso algoritmo di pyTexture ma lavora sui raster binari di GRASS e risulta estremamente più veloce e performante. Anche i sorgenti di questo modulo possono essere richiesti ed ottenuto contattando l'autore.

Bibliografia

Bourke P., 1987 – "Determining if a point lies on the interior of a polygon". Downloaded printed

Cavallini P, Neteler M., 2005 – "I GIS Open Source: un'alternativa possible ?" – MondoGIS n° 47

Cavallini P, Neteler M., 2006 – "L'analisi geografica Open Source: GRASS GIS parte 2 – raster" – MondoGIS n° 52

Neteler M. and H. Mitasova – 2004 – Open Source GIS: A GRASS GIS Approah" - Kluwer Academic Publisher/Springer, Boston, -Second edition ISBN: 1-4020-8064-6

Sanesi G., 2000 - "Elementi di pedologia" - Edagricole, Bologna

Texture AutoLookup (TAL) ver 4.2, 2002 - www.agri.upm.edu.my/~chris/tal.

Van Rossum G. - 2005 - "Manuale di riferimento di python" - Ed. Italiana, Fred L. Drake, Jr., editor

WATTON J. D., 1997 - "Comparing C, Fortran, Lisp, java, python, perl, scheme (guile), and tcl using a floating point numerical test - point inside polygon". Applied mathematics and computer technology center alcoa technical center. Distribution documents.